HEART DISEASE PREDICTION USING RANDOM FOREST ALGORITHM

AIM: To build a predictive model using the Random Forest algorithm to classify and evaluate the likelihood of heart disease in patients based on clinical and lifestyle features, enabling early detection and preventive healthcare interventions.

DESCRIPTION: Heart disease remains one of the leading causes of mortality worldwide. Early detection and accurate prediction are crucial for timely treatment and prevention. This project focuses on developing a machine learning model using the Random Forest algorithm to predict the presence of heart disease based on various medical attributes such as age, sex, blood pressure, cholesterol levels, chest pain type, resting ECG results, and others.

- The Random Forest algorithm, an ensemble learning method, operates by constructing a multitude of decision trees during training and outputting the mode of the classes for classification tasks. It is known for its high accuracy, robustness, and ability to handle missing data and noisy datasets.
- In this project, a publicly available dataset (such as the UCI Heart Disease dataset) is used to train and test the model. Data preprocessing steps including handling missing values, feature selection, and normalization are applied to ensure data quality. The performance of the model is evaluated using metrics such as accuracy, precision, recall, F1-score, and ROC-AUC.

SOURCE CODE:

```
import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

# Correct the import statement to import RandomForestClassifier from sklearn.ensemble

from sklearn.ensemble import RandomForestClassifier
```

```
# Import precision_score, recall_score, f1_score, confusion_matrix and classification_report from sklearn.metrics

from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score, classification_report

df=pd.read_csv('/content/framingham.csv')

df.dropna(axis=0, inplace=True)

X=df.iloc[:,:-1].values

y=df.iloc[:,-1].values

df.head()

df.isnull().sum()

df_1=df[df['TenYearCHD']==1]

df_2=df[df['TenYearCHD']==0]

print(df_1['glucose'].mean()-df_2['glucose'].mean())

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size=0.30,random_state=42)

sc=StandardScaler()

X_train = sc.fit_transform(X_train)

X_test = sc.transform(X_test)

# Ensure the classifier is instantiated as RandomForestClassifier

classifier = RandomForestClassifier(random_state=0)

classifier.fit(X_train,y_train)

y_pred = classifier.predict(X_test)

print(y_pred)

predicted = pd.DataFrame()

predicted['y_test'] = y_test

predicted['y_pred'] = y_pred

print(predicted)

# Calculate and print the accuracy score using the true test labels and the predicted labels

print("Accuracy Score:", accuracy_score(y_test, y_pred))

# Generate and print the classification report which includes precision, recall, f1-score, and support
```

```
print(classification_report(y_test, y_pred))
# Calculate and print the confusion matrix
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
pd.reset_option('all')
```

DATASET:

male	age	education	currentSm	cigsPerDa	BPMeds	prevalent	prevalent	diabetes	totChol	sysBP	diaBP	BMI	heartRate	glucose	TenYearCHD
1	39	4	0	0	0	0	0	0	195	106	70	26.97	80	77	0
0	46	2	0	0	0	0	0	0	250	121	81	28.73	95	76	0
1	48	1	1	20	0	0	0	0	245	127.5	80	25.34	75	70	0
0	61	3	1	30	0	0	1	0	225	150	95	28.58	65	103	1
0	46	3	1	23	0	0	0	0	285	130	84	23.1	85	85	0
0	43	2	0	0	0	0	1	0	228	180	110	30.3	77	99	0
0	63	1	0	0	0	0	0	0	205	138	71	33.11	60	85	1
0	45	2	1	20	0	0	0	0	313	100	71	21.68	79	78	0
1	52	1	0	0	0	0	1	0	260	141.5	89	26.36	76	79	0
1	43	1	1	30	0	0	1	0	225	162	107	23.61	93	88	0
0	50	1	0	0	0	0	0	0	254	133	76	22.91	75	76	0
0	43	2	0	0	0	0	0	0	247	131	88	27.64	72	61	0
1	46	1	1	15	0	0	1	0	294	142	94	26.31	98	64	0
0	41	3	0	0	1	0	1	0	332	124	88	31.31	65	84	0
0	39	2	1	9	0	0	0	0	226	114	64	22.35	85	NA	0
0	38	2	1	20	0	0	1	0	221	140	90	21.35	95	70	1
1	48	3	1	10	0	0	1	0	232	138	90	22.37	64	72	0
0	46	2	1	20	0	0	0	0	291	112	78	23.38	80	89	1
0	38	2	1	5	0	0	0	0	195	122	84.5	23.24	75	78	0
1	41	2	0	0	0	0	0	0	195	139	88	26.88	85	65	0
0	42	2	1	30	0	0	0	0	190	108	70.5	21.59	72	85	0
0	43	1	0	0	0	0	0	0	185	123.5	77.5	29.89	70	NA	0
0	52	1	0	0	0	0	0	0	234	148	78	34.17	70	113	0
0	52	3	1	20	0	0	0	0	215	132	82	25.11	71	75	0
1	44	2	1	30	0	0	1	0	270	137.5	90	21.96	75	83	0
1	47	4	1	20	0	0	0	0	304	102	69	24.18	62	66	1

OUTPUT:

```
8.11229544713271
[0 0 0 ... 0 0 0]
      y_test  y_pred
0          0       0
1          0       0
2          0       0
3          0       0
4          1       0
...      ...     ...
1092       0       0
1093       0       0
1094       0       0
1095       0       0
1096       0       0

[1097 rows x 2 columns]
Accuracy Score: 0.8404740200546946
              precision    recall  f1-score   support

           0       0.85      0.99      0.91       923
           1       0.48      0.06      0.11       174

    accuracy                           0.84      1097
   macro avg       0.66      0.53      0.51      1097
weighted avg       0.79      0.84      0.79      1097

Confusion Matrix:
 [[911  12]
 [163  11]]
```

RESULT:

The Random Forest algorithm was successfully applied to the heart disease dataset to predict the likelihood of heart disease in patients. After preprocessing and splitting the dataset into training and testing sets , the model was trained using multiple decision trees and evaluated for performance.